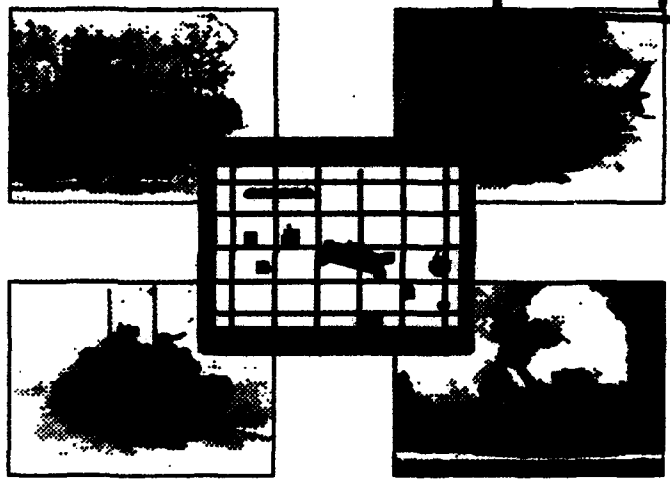
Report No.
ADST/WDL/TR-93-003260

Advanced Distributed Simulation Technology

Digital Voice Gateway Reference Guide



Prepared By:

LORAL

ADST Program Office
12151-A Research Pkwy.
Orlando, FL 32826

Contract No. N61339-91-D-0001-0045

January 31, 1994
REV 2.00

Prepared for:

STRICOM
Simulation Training and Instrumentation Command
Naval Training Systems Center
12350 Research Parkway
Orlando, FL 32826-3275

REPORT DOCUMENTATION PAGE

Form approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	01/28/94	Version 2.00 12/1/93 - 12/19/93

4. TITLE AND SUBTITLE
Digital Voice Gateway Reference Guide

5. FUNDING NUMBERS
Contract No.
N61339-91-D-0001-0045

6. AUTHOR(S)
Van Hook, Dan
Stadler, Ed

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Loral Systems Company
ADST Program Office
12151-A Research Parkway
Orlando, FL 32826-3283

8. PERFORMING ORGANIZATION REPORT NUMBER
ADST/WDL/TR-93-003260

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
STRICOM Simulation Training and Instrumentation Command
Naval Training Systems Center
12350 Research Parkway
Orlando, FL 32826-3275

10. SPONSORING ORGANIZATION REPORT

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

12b. DISTRIBUTION CODE
A

13. ABSTRACT (Maximum 200 words)

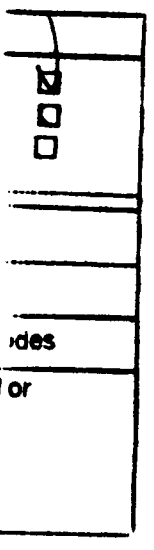
The Digital Voice Gateway (referred to as the 'DVG' in this document) transmits and receives four full duplex encoded speech channels over the Ethernet. The information in this document applies only to DVGs running firmware of the version listed on the title page. This document, previously named Digital Voice Gateway Reference Guide, BBN Systems and Technologies Corporation, Cambridge, MA 02138, was revised for revision 2.00. This new revision changes the network protocol used by the DVG, to comply with the SINCGARS radio simulation (For SIMNET 6.6.1). Because of the extensive changes to revision 2.00 a separate document was created rather than supplying change pages.

14. SUBJECT TERMS

15. NUMBER OF PAGES

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL



1. General Information

The Digital Voice Gateway (referred to as the 'DVG' in this document) transmits and receives four full duplex encoded speech channels over the Ethernet. The information in this document applies only to DVGs running firmware of the version listed on the title page.

This document, previously named Digital Voice Gateway Reference Guide, BBN Systems and Technologies Corporation, Cambridge, MA 02138, was revised for revision 2.00. This new revision changes the network protocol used by the DVG, to comply with the SINCGARS radio simulation (For SIMNET 6.6.1). Because of the extensive changes to revision 2.00 a separate document was created rather than supplying change pages.

1.1. What Is Covered In This Guide

This guide presents an overview of the DVG hardware and software to a level of detail such that the reader can understand the types of processing performed by the DVG as well as debug many common system problems. It also presents how to configure the hardware and software components of the DVG. This guide does not address the incorporation of the DVG into larger systems or connection to specific sources of voice such as CB radios or other audio equipment.

1.2. Related Documentation

The SIMVAD speech coding board is described in the following two BBN STC reports:

SIMVAD Hardware Documentation

SIMVAD System Real-Time Software Documentation

Also, the following Motorola publications are useful in working with the MVME147 CPU hardware and its firmware debugger, 147Bug:

MVME147 MPU and MVME712 Transition Module User's Manual

MVME147Bug User's Manual

The DVG version 2.00 uses a subset of the SINCGARS simulation protocol (which will be discussed briefly in the section entitled Protocol Format). This protocol is detailed in the following document.

> SIMNET SIMULATION OF RADIO COMMUNICATION: A Testbed for Investigation Combat Vehicle C3I Technology. Report No. 7352 (SIMNET LIBRARY serial # 50074)

2. System Description

The DVG consists of a high performance, high level of integration CPU mated with special purpose speech coding hardware in an industry standard VMEbus enclosure. The software that controls the DVG resides in EPROMs on the CPU board.

An overview of the DVG hardware and firmware will be presented in this section.

2.1. Hardware Description

2.1.1. MVME147 CPU

The Motorola MVME147 CPU controls the DVG. It performs all network input and output, controls and initializes the SIMVAD speech coding boards, and interfaces to the DVG terminal port. It has the following features:

MC68030 microprocessor running at 25 MHz

MC68882 floating-point processor

4 MB of dynamic RAM expandable to 16 MB

4 serial ports

SCSI interface

time of day clock

2 KB of battery backed RAM

debugger in ROM

VMEbus system controller

reset and abort switches

printer port

Ethernet interface

2.1.2. MVME712 Transition Module

The Motorola MVME712M Transition Module provides the input/output connections for the MVME147 CPU. It connects to the MVME147 via an adapter board plugged into the P2 VMEbus connector of the slot that the MVME147 is plugged into. The MVME712M is mounted on the rear of the DVG chassis and provides connections for the Ethernet transceiver cable and the DVG terminal port. Three other serial ports, a SCSI port and a printer port on the MVME712M are unused by the DVG.

2.1.3. SIMVAD Speech Coding Board

The SIMVAD speech coding board, developed and manufactured by BBN STC, consists of two complete and independent full duplex speech coding channels on a single 6U VMEbus module. The standard DVG supports four voice channels and hence contains two SIMVAD boards. Each channel consists of the following components:

TMS320C25 signal processor

32 KW data RAM

32 KW program RAM

16 KW program ROM

A/D and D/A converter

512 Words each of input and output fifo

Each channel uses the APCHQ coding algorithm (in firmware) to encode and decode speech at a 16 Kbit per second or 32Kbit per second rate. As used in the DVG, the SIMVAD supports an input voltage range of +/- 3 volts, differential and an output range of +/- 6 volts, differential.

2.1.4. VMEbus Chassis

The DVG chassis is a 12-slot, 6U, rack mountable unit with integral fans manufactured by Mupac. The power supply is rated at 400 Watts and provides the following outputs:

Voltage (Volts)	Maximum Current (Amps)
+5	70
+12	10
-12	10
-5	2

Input power is via a standard 15 Amp, single phase, 120 Volt, three wire power cord. The DVG draws approximately 3 Amps at 120 Volts when operating.

2.2. Firmware Description

The DVG firmware runs as a single program in complete control of the MVME147 CPU. All network and voice channel processing is performed in the context interrupts generated by a selected SIMVAD voice coding board. Terminal I/O is handled in the foreground using system functions provided by 147Bug, the MVME147's EPROM-resident debugger. This section will describe in some detail the functions performed by the firmware.

2.2.1. Boot-up and Initialization

The DVG firmware is designed to perform its initialization tasks from power-up or front panel reset and enter its main processing loop without user intervention. Booting takes about 10 seconds. For debugging and configuration purposes, however, it is important to be familiar with the sequence of events that lead from a reset to full operation.

The typical output from the DVG console port after a reset or power cycle is shown below.

```
Copyright Motorola Inc. 1988, All Rights Reserved
VME147 Monitor/Debugger Release 2.0 - 3/16/89
FPC passed test
MMU passed test
COLD Start
147-Bug> Searching for ROM Boot
147-Bug>G FFA0000E
Effective address: FFA0000E
Digital Voice Gateway
version 2.0.0 of Thu Nov 16 15:36:47 EST 1993
LORAL Systems Corporation
Orlando, FL 32826
Initializing timers.
Initializing clock interrupt.
Initializing network.
Ethernet Address: 08003e204674
Initializing SIMVAD cards.
channel0: found at 0xe000.
channel1: found at 0xe000.
channel2: found at 0xe100.
channel3: found at 0xe100.
Initializing keyboard.
VG>
```

The DVG firmware receives control from the 147Bug debugger following power-up or front panel reset. The firmware first copies itself from address 0xFFA00000 (in EPROM) to address 0x6000 (in RAM), the address it was linked to run at. Control is then transferred to the firmware copy at address 0x6000 and initialization continues. The firmware runs out of RAM because memory accesses to RAM are faster than those to EPROM on the MVME147 CPU.

After printing out a banner, the firmware proceeds to initialize its various subsystems: timers, interrupts, network, SIMVADs and keyboard. Error messages are printed should initialization fail and control will be returned to 147Bug in the event of a fatal error.

During network initialization, the Ethernet address programmed into a PROM on the MVME147 CPU is displayed as shown in the above printout. This address will of course be different for each DVG.

During SIMVAD initialization, each pair of channels (corresponding to a single SIMVAD board) is checked for existence and a message is printed as shown below for each channel found. Each SIMVAD board is then reset (its two red LEDs blink once together when this operation is performed) and the diagnostic status returned by the SIMVAD firmware is checked by the DVG firmware. The possible diagnostic messages that could be printed are:

svN: diagnostics: TMS Operation Error

svN: diagnostics: EPROM Error

svN: diagnostics: Program RAM Error

svN: diagnostics: Data RAM Error

svN: diagnostics: D/A-A/D Test Error

The particular voice channel that failed is indicated by 'svN', where N is the channel number. The first four types of failures (bad TMS 320C25, bad EPROM checksum, and RAM errors) indicate a bad SIMVAD board. The fifth error (D/A-A/D Test Error), however, can occur without a board being bad. The D/A-A/D test is performed by outputting a triangle wave on the D/A output and comparing it with the signal measured on the A/D input. If the difference is too great, the 'D/A-A/D Test Error' is reported. This error can occur if there is too much noise pickup on the cables connected to the SIMVAD boards as well as if the analog input circuitry on the SIMVAD is bad, so its occurrence should be regarded with suspicion. A good test is to disconnect the input cable and reset the DVG again - if the error persists, then the SIMVAD should be suspected.

Following SIMVAD initialization, the lowest numbered voice channel found (generally number 0) is initialized to generate an interrupt at the SIMVAD frame rate (26.25 milliseconds). The service routine for this interrupt handles all processing for the voice channels as well as the higher level network input/output processing as described in later sections.

Finally, the terminal interface is initialized and the DVG prompt ('VG>') is printed on the console port. The DVG firmware then enters its main processing loop and begins to service the voice channels and the network.

2.2.2. Protocol Format

A brief discussion of the SINCGARS protocol follows. It is provided so that the reader may understand the operations of the DVG software.

Version 2.00 of the DVG uses a subset of the SIMNET SINCGARS simulation protocol. Mainly DVG 2.00 does not use a receiver PDU, which is implemented in SINCGARS simulation. The DVG does not incorporate any terrain modeling for signal degradation. The DVG does use the remaining transmitter and signal PDUS. To conform to the SINCGARS radio simulation the DVG produces one transmitter PDU, for each radio it is simulating, every 5 seconds. The transmitter PDU contains information including: radio id, location, frequency, and state of the transmitter (and whether it is a change from the last transmission). Signal PDUs are produced whenever valid speech data is acquired from the SIMVAD boards. The signal PDUs contain information including: radio id and coded speech data.

A transmitter PDU is sent whenever the state of the transmitting radio has changed or 5 seconds has elapsed. Signal PDUs are sent whenever valid data is received from a SIMVAD board. The SIMVAD boards produce data in 56 byte buffers, one buffer makes up the data portion of the signal PDU. Signal PDUs are sent, until the SIMVAD board supplying the buffers, no longer produces valid data.

2.2.3. Network Service

The network is serviced from within the handler for the frame interrupt set up as described in the 'Boot-up and Initialization' section. This module drains the queue of received packets on each invocation. It discards all packets that do not correspond to the correct protocols (0x86 and 0x89), or the PDU kind is not a Transmitter or Signal PDU, or the packet is from a different exercise. (Exercise ID will be discussed later) A buffer is allocated (from a pool of free buffers) for each received packet.

If the packet contains a transmitter PDU, the frequency information within this PDU is stored in a hash table (the hash value is obtained using the radio id). When a signal PDU is received, a search is made in the hash table using the radio id obtained within the signal PDU. If no match is found the packet is discarded. If a match is found a header is set up to point to the speech data within the PDU. The headers are passed to the SIMVAD software module for transmission to the appropriate SIMVAD card. A use count is maintained in the buffer so that when the SIMVAD software module has freed all its accesses to the buffer (i.e., the headers), the buffer can be returned to the free pool.

Transmission of signal and transmitter packets over the network occurs as a result of calls from the SIMVAD software module into the network software module. The SIMVAD software module places speech frames into a buffer allocated from the free

pool by the network software module. An Ethernet header is prepended to the buffer and the packet is transmitted over the network when the maximum number of speech frames per packet is reached (currently the maximum is 1), when the maximum Ethernet packet size if reached, or when a 'flush' function is called by the SIMVAD software module (always called at the end of the frame interrupt to insure that all buffered data is transmitted with only small latency). All packets transmitted over the network are also turned around and processed by the receiving software to allow channels in loopback mode to receive their own frames back.

2.2.4. SIMVAD Service

The SIMVAD voice channels are serviced from within the context of the frame interrupt set up as described in the section 'Boot-up and Initialization'. The frame queue on each SIMVAD is drained and the frames are placed in a buffer allocated by the network software module and transmitted by the network software module. When a frame of data is received from a SIMVAD board, a check is made to determine if the data is valid or just noise created by the SIMVAD board or a signal that does not reach the set threshold level. If the data is valid further checks are made to determine the state of the transmitter. If a new state is indicated a new transmitter PDU will be sent. Logic in the software determines if the signal PDU is the last of its transmission. If so the sync field within the PDU indicates this by the End Of Message sync, if not the sync field is normal. Also, timing routines which send transmitter PDUs every 5 seconds are maintained, and a group of signal PDUs will be interrupted to send a new transmitter PDU if necessary.

The SIMVAD software module receives headers that point at speech frames within a network packet in a buffer. It will en queue these frames on the appropriate SIMVAD card after introducing a frame of delay for the first frame of a sequence of frames in an attempt to reduce the probability of an anomaly due to variable transmission delays for the packets containing an utterance.

The SIMVAD software module attempts to "lock on" to a particular source of speech so as to eliminate the anomalies due to interleaving the speech frames from more than one source. The algorithm is essentially to maintain a state variable for each channel that can be either 'idle' or 'playing'. If the channel is 'idle', the first frame destined for a particular channel is played, the source is noted and the state is set to 'playing'. As long as frames continue to be received from the same source, they are played and the state is maintained as 'playing'. Frames destined for the channel from other sources are discarded. When no more frames from the source that has been "locked onto" are received, the state is set back to 'idle' and once again all sources can "compete" for the channel.

2.2.5. Ethernet Speech Packet Format

The Ethernet packets generated by the DVG conform to the Radio PDUs in the SINCGARS Radio Simulation. Of those, the transmitter and signal PDUs are utilized (the receiver PDU is not implemented on the DVG).

The Ethernet packet has the format shown below. The default Destination Address for transmitter and signal packets are as follow: Transmitter 030000000086, and signal 030000000089 (hex). The Ethernet type is 21000 for both transmitter and signal packets. The Source Address is obtained from the ROM on the CPU board. Byte offsets from the start of the packet are shown for each component.

Byte count starts with 1.

802.2/802.3 packets:

MAC	DESTINATION 6		
	SOURCE 6		
	LENGTH 2		
SAP	DSAP 1	SSAP 1	CNTL 1
SNAP	ORG_CODE 3	ETHER_TYPE 2	
	DATA 46-1492		
	CRC 4		

LENGTH is the number of bytes following the MAC header (the DESTINATION, SOURCE, LENGTH).

SAP (Service Access Point) header specifies ports on the source and destination machines and the type of service.

DSAP is the destination service access point.

SSAP is the source service access point.

CNTL is the type of service (we're using 3 which is "unnumbered service").

The SNAP (Sub-Network Access Point) header is present only if DSAP == SSAP == 0xAA == 170 (this is what we do).

ORG_CODE is an organization code (we're using 0).

ETHER_TYPE is the type of the encapsulated protocol (in our case, 21000).

The data link layer (in the ISO protocol stack) is split into two sublayers:

LLC == Link Layer control

MAC == Media Access Control

The new format is shown below in terms of C language structures:

```
/* An Ethernet header (version2.0, not 802.3): */
typedef struct {
  NetworkAddress to;
  NetworkAddress from;
  short type;
} NetworkHeader;
/* An Ethernet header (802.3, not version2.0): */
typedef struct {
  NetworkAddress to;
  NetworkAddress from;
  short length;
} NetworkHeader8023;
/* An Ethernet packet (version2.0): */
typedef struct {
  NetworkHeader hdr;
  char data[NETWORK_BUFFER_SIZE];
} NetworkBuffer;
/* An Ethernet packet (802.3): */
typedef struct {
  union {
    NetworkHeader hdr2;
    NetworkHeader8023 hdr8023;
  } h;
  char data[NETWORK_BUFFER_SIZE];
} NetworkPacket;
```

Notice that the structures are set up so that it is easy to implement DVG that will accept both the version2.0 packets and 8023 packets. First, check length in the header and if it is larger than maximum packet length then it is version2.0 (version2.0 saves type in this field). However, there is no foreseeable need in handling version2.0 packets, so that 1.43 and 2.00 DVG does not implement this feature. In 8023 format

DSAP and SSAP are stored before the data, there are macros to get/set DSAP, SSAP and data information. Following are the C macros:

```
#define GET_DSAP(p)       (((unsigned char *)(p))[0])
#define GET_SSAP(p)       (((unsigned char *)(p))[1])
#define GET_CONTROL(p)    (((unsigned char *)(p))[2])
#define GET_PROTOID(p)    ((((unsigned char *)(p))[3] << 16) |
                          (((unsigned char *)(p))[4] <<  8) |
                          ((unsigned char *)(p))[5])
#define GET_ETHER_TYPE(p) (((unsigned short *)(p))[3])
#define GET_DATA_PTR(p)   ((unsigned char *)(p) + 8)
#define SET_DSAP(p,x)     (((unsigned char *)(p))[0] = (x))
#define SET_SSAP(p,x)     (((unsigned char *)(p))[1] = (x))
#define SET_CONTROL(p,x)  (((unsigned char *)(p))[2] = (x))
#define SET_PROTOID(p,x)  ((unsigned char *)(p))[3] = (x) >> 16;
                          ((unsigned char *)(p))[4] = (x) >>  8;
                          ((unsigned char *)(p))[5] = (x)
#define SET_ETHER_TYPE(p,x) ((unsigned short *)(p))[3] = (x)
```

2.2.6. Terminal Interface

The DVG supports a terminal interface used for debugging, development and configuration. From this interface, various system and performance parameters may be entered and/or examined. The DVG terminal interface may be accessed via the CONSOLE port 1 connector at the rear of the unit. See the section on 'Hardware Configuration' details.

The terminal interface is command line oriented, i.e., the user types commands and parameters in response to a prompt. Each of the available commands and their resulting output will be illustrated and described in the following pages.

The command interface supports simple line editing via sequences of control characters, a list of these folow:

^p previous line

^n next line

^b back one character

^f forward one character

^d delete character under cursor

^a go to beginning of line

^e go to end of line

^s delete previous character

Commands and parameters are separated by spaces. Entire commands need not be typed (although all examples below will use fully typed-out commands) only enough letters to differentiate the command from all others at that point in the parsing must be typed. Help about the possible choices at any point may be obtained by typing a '?'. Typing '?' immediately after a command gives a single line of help about that command while typing '?' after a space or at the beginning of a line gives all the possible choices at that point. In all the examples that follow, the DVG prompt is displayed as 'VG>'.

2.2.6.1. Top Level

The top level of commands may be viewed by typing '?' as show below:

```
VG> ?
 Commands:
voice       -voice channel commands
radio       -radio commands
network     -network commands
system      -system commands
config      -configuration commands
exit        -exit program
quit        -exit program
help        -parser editor help
```

Each of these commands or command groups will be explained in the sections that follow.

2.2.6.2. Voice Command Group

The voice command group contains commands related to the voice input and output channels. The voice commands may be viewed by typing a '?' after entering the 'voice' command (followed by a space), as shown below:

```
VG> voice ?

 Voice Channel Commands:

loopback        -set or show loopback status
pushtotalk      -set or show pushtotalk check status
speechvalid     -set or show speech level check status
threshold       -set or show threshold
rate            -set or show coding rate
activity        -set or show on/off status
mapping         -set or show frequency to physical channel mapping
statistics      -show statistics
zerostatistics  -zero statistics
restart         -restart simvads
```

2.2.6.2.1. Loopback

The voice channels may be placed in loopback mode (input connected to output) via the 'loopback' command. In this mode, all incoming speech frames are output on the same channel that they were input from. A single channel may be put into loopback mode by entering the line

```
VG> voice loopback set N on
```

where N is the channel number to put into loopback mode. Likewise, loopback mode may be turned off for a single channel by entering the line

```
VG> voice loopback set N off
```

where N, is again, the channel number to take out of loopback mode. All channels may be put into or taken out of loopback mode by (respectively) typing the lines

```
VG> voice loopback all on
```

```
VG> voice loopback all off
```

Finally, the loopback status of each channel may be displayed as follows. The display shown below is for the case where loopback is off for all the channels.

VG> voice loopback show

channel	on/off
0	OFF
1	OFF
2	OFF
3	OFF

The loopback is accomplished in software - no hardware connection is made. All speech frames input from the channels in loopback are still directed to the network, however, and all speech frames from the network that are directed to channels in loopback mode are discarded, i.e., they are not output.

Loopback mode is particularly useful for debugging. The functionality of a particular voice channel may be verified by placing the channel in loopback mode, driving the input with a known wave form, and attempting to observe the resulting wave form (after coding, passing through the internal software path and then decoding) at the outputs on the same channel. Also, loopback mode may be used in conjunction with the voice statistics (see below) command selection to verify that the system is passing frames from input to output without depending on a working network connection.

2.2.6.2.2. Pushtotalk

The 'pushtotalk' command allows push-to-talk checking to be enabled or disabled for the voice channels. The default condition is for push-to-talk checking to be enabled, i.e., if the push-to-talk input for a channel is asserted, an input frame (read from the local channel) will be passed to the network while if it is deasserted, the frame will be discarded.

Push-to-talk checking may be turned on or off for a single channel by typing the commands (respectively)

VG> voice pushtotalk set N on

VG> voice pushtotalk set N off

where N is the channel to affect. Checking may be turned on or off for all channels by typing the commands (respectively)

VG> voice pushtotalk all on

VG> voice pushtotalk all off

Finally, the status of push-to-talk checking may be displayed as follows. The display shown below is for the case of checking turned on for all channels.

VG> voice pushtotalk show

channel	on/off
0	ON
1	ON
2	ON
3	ON

The SIMVAD voice coding boards' push-to-talk inputs naturally float to an asserted level. This results in a flood of packets on the network for any channel that has floating inputs since it appears to the DVG software that the push-to-talk input is always asserted. This command is useful in debugging because it can be used to force the speech frames from a particular voice channel to be placed on the network independent of the state of the channel's push-to-talk input. This can be an aid in isolating a fault to push-to-talk circuitry connected to a DVG voice channel.

2.2.6.2.3. Speechvalid

The 'speechvalid' command enables checking of the speech-valid bit in each frame of speech read from the SIMVAD speech coding board. The SIMVAD tests a measure of the signal power in each frame against a threshold and sets the valid bit if the power exceeds the threshold. Otherwise, the SIMVAD clears the valid bit. The default threshold (which may be set via the 'threshold' command) is 0 so that all speech will be considered valid unless the threshold is raised.

Speech-valid checking may be turned on or off for a voice channel by typing the commands (respectively)

VG> voice speechvalid set N on

VG> voice speechvalid set N off

where N is a voice channel number. Speech-valid checking may be turned on or off for all channels by typing (respectively)

VG> voice speechvalid all on

VG> voice speechvalid all off

Finally, the speech-valid checking status for each channel may be displayed as follows. The example below is for the case of speech-valid checking turned off for all channels.

VG> voice speechvalid show

channel	on/off
0	OFF
1	OFF
2	OFF
3	OFF

Speech-valid checking is turned off by default for all channels. It may be useful for some applications that wish to send only speech frames that contain valid (as determined by sufficient power) speech.

2.2.6.2.4. Threshold

The threshold for speech-valid checking may be entered via the 'threshold' command. The default threshold is 0 for all channels. The threshold is used by the SIMVAD in determining whether the input speech in a frame is valid, i.e., has sufficient energy to be considered signal instead of mere noise. A threshold may be entered for a voice channel typing the command

VG> voice threshold set N THRESHOLD

where N is a voice channel number and THRESHOLD is an unsigned hex number. A threshold may be entered for all channels at the same time by typing the command

VG> voice threshold all THRESHOLD

where THRESHOLD is an unsigned hex number. Finally, the threshold for each channel may be displayed as follows. The example below is for the case of default thresholds for all channels.

VG> voice threshold show

channel	threshold
0	0x00000000
1	0x00000000
2	0x00000000
3	0x00000000

2.2.6.2.5. Rate

The 'rate' command allows setting of the coding rate of the SIMVAD boards. The rate will be either 16 or 32 Kbps APCHQ coding. The rate may be set for one or all channels. The 'rate' command will also show the current rates for all channels. The following example shows the format for the 'rate' command.

VG> voice rate show

channel	rate
0	16Kbps
1	16Kbps
2	16Kbps
3	16Kbps

VG> rate set 2 16 (Sets SIMVAD channel 2 to 16 Kbps APCHQ)

VG> rate all 16 (Sets all SIMVAD channels to 16 Kbps APCHQ)

2.2.6.2.6. Activity

Channels may be turned on or off individually or all at once. In previous releases, the only way to disable a channel was to pull out the corresponding board or to ensure that the push-to-talk input was disabled. As an example, channel 0 may be turned off by typing "voice activity set 0 off" The channel number must be a physical (device) number. An example of the activity function follows.

VG> voice activity show

device	frequency	status
sv0	33000KHz	ON
sv1	43000KHz	ON
sv2	53000KHz	ON
sv3	63000KHz	ON

All channels may be turned on or off at the same time using the 'activity all "on/off"' command. One channel may be turned on or off by typing 'activity set N "on/off"' command where N represents the SIMVAD channel number.

2.2.6.2.7. Mapping

In the SINCGARS radio protocol, a radio determines whether to listen to another radios transmissions based upon the frequency settings which are contained in the transmitter PDU. A frequency can be mapped to a physical SIMVAD channel. When transmissions from other radios set to the same frequency are processed, there signal data will be routed to that particular SIMVAD channel. The mapping command will show the current settings as shown below.

VG> voice mapping show

device	address	frequency
sv0:	0xe00042	33000 KHz
sv1:	0xe00043	43000 KHz
sv2:	0xe10044	53000 KHz
sv3:	0xe10045	63000 KHz

VG>

The mapping may be set by use of the "set" command in the voice mapping menu. The example below sets the "A" port on simvad device 2 to be a frequency of 44000 KHz.

VG> voice mapping set 2 44000

VG>

VG> voice mapping show

device	address	frequency
sv0:	0xe00042	33000 KHz
sv1:	0xe00043	43000 KHz
sv2:	0xe10044	44000 KHz
sv3:	0xe10045	63000 KHz

VG>

2.2.6.2.8. Statistics

The firmware will automatically restart all the SIMVAD cards in the system and print a message to that effect on the console upon detection of a failure or bad data from a SIMVAD. The count of the number of times an error has been detected for each SIMVAD channel is displayed with the other voice statistics as shown below:

VG> voice statistics

frames input:

chan	total	fr/1s	fr/10s	fr/60s
0:	105346	38	38	16
1:	105345	38	38	16
2:	105347	38	38	16
3:	1704	0	0	0

frames output:

chan	total	fr/1s	fr/10s	fr/60s
0:	53732	0	0	0
1:	56123	0	0	0
2:	0	0	0	0
3:	832	0	0	0

IO errors:

chan	total
0:	0
1:	0
2:	0
3:	0

This table shows the total speech frames input (read) from each channel, output (written) to each channel, and the running average frame rates input and output averaged over 1 second, 10 seconds and 60 seconds for each channel. Input frames are sent out over the network, while output frames are received from the network (or looped back). In the example above, all four channels are generating frames at their maximum rate (a little less than 40 per second) while only channels 0 and 1 are receiving frames. Also, the DVG in the above example has been running for less than 60 seconds so that the 60 second running averages are not accurate. Another caveat is that the input frames statistic is incremented only if the push-to-talk and speech-valid tests (see above) are successful.

The total number of resets that have occurred is the sum of the entries for all the channels in the IO errors section. The SIMVADs may also be restarted from the console by typing "voice restart".

This command is useful for debugging in that it provides information about whether frames from a particular voice channel are being output after being received from the network. Also, a channel can be put into loopback mode to verify that speech frames are being read from it and written out to it (see section on 'loopback' above).

2.2.6.2.9. Zerostatistics

The voice statistics may be zeroed by typing the command

VG> voice zerostatistics

2.2.6.3. Radio Command Group

The radio command group contains commands related to radio identification. The radio commands may be viewed by typing a '?' after entering the 'radio' command (followed by a space), as shown below:

```
VG> radio ?

  Radio Commands:

set         -set radio site host vehicle and radio numbers

show        -show radio parameters for one radio

all         -show radio parameters for all channels
```

2.2.6.3.1. Set

The 'set' command sets the radios site, host, vehicle and radio numbers. Each radio must have a unique id, which is comprised of site, host, vehicle and radio number. These parameters are set by assigning these numbers to a particular SIMVAD channel as shown in the example below.

```
VG> radio set 0 2 54 1 0

VG>
```

(This sets SIMVAD channel 0 to a site of 2, host of 54, a vehicle of 1 and radio 0*.)

* A radio number of 0 corresponds to a standalone radio in the SINCGARS simulation.

2.2.6.3.2. Show

The 'show' command shows the current set up for one radio specified. An example of the 'show' command follows.

```
VG> radio show 0

        Radio 0 Parameters

        Site ID is          2       Host ID is          54

        Vehicle ID is       1       Radio ID is         0

        Radio Frequency is          44000

VG>
```

2.2.6.3.3. All

The 'all' command shows the current set up for all radios configured in the system. An example of the 'all' commands output is shown below.

```
VG> radio all
        Radio 0 Parameters
        Site ID is          2       Host ID is      54
        Vehicle ID is       1       Radio ID is     0
        Radio Frequency is          44000

        Radio 1 Parameters
        Site ID is          2       Host ID is      54
        Vehicle ID is       2       Radio ID is     0
        Radio Frequency is          43000

        Radio 2 Parameters
        Site ID is          2       Host ID is      54
        Vehicle ID is       3       Radio ID is     0
        Radio Frequency is          45000

        Radio 3 Parameters
        Site ID is          2       Host ID is      54
        Vehicle ID is       4       Radio ID is     0
        Radio Frequency is          46000
VG>
```

2.2.6.4. Network Command Group

The network command group contains commands related to network input and output. The network commands may be viewed by typing a '?' after entering the 'network' command (followed by a space), as shown below:

```
VG> network ?
 Network Commands:
voiceaddress          -show ethernet address for voice traffic
setvoiceaddress       -set ethernet address to for voice traffic
myaddress             -display ethernet address of this station
getstats              -show low-level network statistics
zerostats             -zero network statistics
packetstats           -show packet statistics
zeropacketstats       -zero packet statistics
bufferstats           -show buffer statistics
zerobufferstats       -zero buffer statistics
setaggregation        -set max pdus per packet
aggregation           -show max pdus per packet
settype               -set Ethernet packet type
type                  -show Ethernet packet type
exerciseid            -show exercise id
setexerciseid         -set exercise id
```

2.2.6.4.1. Voiceaddress

The 'voiceaddress' command displays the current Ethernet address that the transmitter and signal packets are being sent to and received from, as shown below.

```
VG> network voiceaddress

        transmitter address:    03 00 00 00 00 89

        signal address:         03 00 00 00 00 86
```

The voice address is specified as six hexadecimal bytes separated by spaces. The bytes are transmitted in the order specified, with bits within a byte transmitted starting from the least significant. This is configured in software, and the user cannot alter its value.

2.2.6.4.2. Setvoiceaddress

The 'setvoiceaddress' command is a carry over from previous versions of the DVG. The 'setvoiceaddress' command will print a message to the screen as shown below:

```
VG> network setvoiceaddress

        The Transmitter PDU and Signal PDU destination
        addresses are configured automatically.
```

2.2.6.4.3. Myaddress

The 'myaddress' command displays the Ethernet address of the DVG as shown below.

```
VG> network myaddress
        ethernet address:    08 00 3e 20 46 74
```

The address is obtained from a ROM on the DVG CPU card.

2.2.6.4.4. Getstats

The command 'getstats' displays statistics from the Ethernet driver software. A typical display produced by this command is shown below.

```
VG> network getstats
successful transmissions                1296740
multiple retries reported               1
single retries reported                 4
retry failures reported                 0
deferrals reported                      125
buffer errors                           0
silo underruns                          0
late collisions                         0
carrier losses                          0
babbling transmitter errors             0
collisions errors                       1280633
memory errors                           0
packets received                        970076
missed packets reported                 0
CRC errors reported                     0
framing errors                          0
silo overruns                           0
stp bit missing                         0
enp bit missing                         0
lance errors                            0
non-SIMNET packets                      0
packets filtered                        0
packets missed - table locked           0
packets missed - index out of range     0
```

This command is especially useful for diagnosing physical Ethernet problems such as loose cables. The 'successful transmissions' statistic should increment as SINCGARS packets are successfully sent over the network by the DVG. Likewise, the 'packets received' statistic should increment as packets of any type are received from the network by the DVG. Physical Ethernet problems are indicated if any of the following statistics are incremented:

retry failures reported
late collisions
carrier losses
babbling transmitter errors
CRC errors reported
framing errors

2.2.6.4.5. Zerostats

The 'zerostats' command zeros the low-level Ethernet statistics displayed by the 'getstats' command. It is invoked as shown below.

VG> network zerostats

2.2.6.4.6. Packetstats

The 'packetstats' command displays network packet statistics of the DVG as shown below.

VG> network packetstats

packet statistics:

	total	pps 1s	pps 10s	pps 60s
signal received	6580	78	82	80
transmitter received	6580	78	82	80
signal sent	6580	78	82	80
transmitter sent	6580	78	82	80
total received	6580	78	82	80
total sent	6580	78	82	80
bad pkts	0	0	0	0

Totals for all packets received and sent and for transmitter and signal packets received and sent are displayed, along with running averages for each category averaged over 1 second, 10 seconds and 60 seconds. Each voice channel will average a little less than 40 packets per second while active so that a four channel DVG can generate a maximum of about 160 packets per second. This command is useful for debugging since it provides an indication of whether a DVG is generating or receiving voice and other network packets.

2.2.6.4.7. Zeropacketstats

The 'zeropacketstats' command zeros all the counters used for the 'packetstats' command. It is invoked as shown below.

VG> network zeropacketstats

2.2.6.4.8. Bufferstats

The 'bufferstats' command displays the status of the DVG's internal buffering system as shown below.

VG> net bufferstats

buffer statistics:

total buffers	140
buffers in use	8
max buffers used	9
times out of buffers	0
total headers	192
headers in use	8
max headers used	9
times out of headers	0

This command is most useful during develo; ment. If, however, the following fields are ever non-zero

times out of buffers

times out of headers

or if the 'max buffers used' field approaches the 'total buffers' field, or if the 'max headers used' field approaches the 'total headers' field, the DVG developers should be notified.

2.2.6.4.9. Zerobufferstats

The 'zerobufferstats' command zeros the counters used by the 'bufferstats' command. It is invoked as shown below.

VG> network zerobufferstats

2.2.6.4.10. Setaggregation

The 'setaggregation' command sets the maximum number of speech frames aggregated into a single Ethernet packet, as shown below.

VG> network setaggregation 4

The above example sets the maximum to 4 frames per packet. The current value of the maximum frames per packet may be displayed using the 'aggregation' command. The setaggregation command will allow setting of a number but has no effect on aggregation for version 2.00 (It is always 1).

2.2.6.4.11. Aggregation

The 'aggregation' command shows the current maximum number of speech frames aggregated into a single Ethernet packet as shown below.

VG> network aggregation

max pdus per packet = 1.

2.2.6.4.12. Settype

The 'settype' command sets the Ethernet packet type. Currently the DVG uses the PROTOCOL_SIM type which is 21000. The settype command is demonstrated below.

VG> network settype 21000

VG>

2.2.6.4.13. Type

The 'type' command displays the current setting of the Ethernet packet type. This will be 21000 by default for the DVG. This command is demonstrated below.

VG> network type

VG> Ethernet type = 21000 = 0x5208

2.2.6.4.14. Exerciseid

The 'exerciseid' command displays the current setting of the exercise id for the simulation. The DVG currently supports one exercise id. This command is demonstrated below.

2.2.6.4.15. Setexerciseid

The 'setexerciseid' sets the exercise id for the current simulation. This is demonstrated in the example below.

```
VG> network setexerciseid 2
VG>
```

2.2.6.5. System Command Group

The system command group consists of commands that are of a general system nature. The system commands may be viewed by typing a '?' after entering the 'system' command (followed by a space), as shown below:

```
VG> system ?

        System Commands:

version     -version and creation date

timing      -show system timing

debug       -debug commands
```

2.2.6.5.1. Version

The 'version' command displays the version number and creation date of the DVG firmware as shown below.

```
VG> system version

        version 2.00 of Thu Nov 16 15:36:47 EST 1993
```

2.2.6.5.2. Timing

The 'timing' command displays system timings as shown below.

```
VG> system timing
```

	min	ave	max (ms)
total interrupt time	2.00	3.66	6.00
network_service()	0.00	1.23	3.00
simvads_service()	1.00	2.38	4.00

The minimum, average and maximum times (in milliseconds) are displayed for the master interrupt, the network service time and the SIMVAD service time.

2.2.6.6. Configuration Command Group

Configuration Control Commands identical to those typed to the console interface may be stored in the battery-backed ram. These commands can be entered

once and will then be automatically executed after rebooting or power cycling. The config entry in the top level menu, supports management of the contents of the battery-backed ram. The config menu is shown below.

```
VG> config ?
Configuration Commands:
delete      -delete line
exec        -execute configuration
init        -initialize (erase) battery-backed RAM
insert      -insert line
list        -list lines
location    -set memory address of config buffer
```

2.2.6.6.1. List

The 'list' command lists the contents of the battery backed ram.

2.2.6.6.2. Insert

A line may be inserted into battery backed ram using the 'insert' command followed by a line number and the commands desired enclosed in double quote marks "". The following shows an example of the contents of the battery backed ram before and after inserting a line to change the Ethernet type code used for DVG packets. The numbers to the left of each line followed by a ":" are line numbers.

```
VG> config list
   0: network setaggregation 1
   1: voice mapping set 0 44000
   2: voice mapping set 1 43000
   3: voice mapping set 2 44000
   4: voice mapping set 3 45000
VG> config insert 1 "network settype 21000"
VG> config list
   0: network setaggregation 1
   1: network settype 21000
   2: voice mapping set 0 44000
   3: voice mapping set 1 43000
   4: voice mapping set 2 44000
   5: voice mapping set 3 45000
VG>
```

2.2.6.6.3. Delete

A line may be deleted by typing "config delete N", where N is the line number to delete.

2.2.6.6.4. Init

The battery-backed ram may be erased by typing "config init".

2.2.6.6.5. Exec

The current configuration may be executed by typing "config exec".

2.2.6.6.6. Location

The memory location of the battery backed ram may be specified using the 'location' command followed by a memory location.

2.2.6.6.7. Exit Command

The 'exit' command causes the DVG firmware to return to the ROM monitor, 147Bug, as shown below.

```
VG> exit
147-Bug>
```

All voice processing ceases upon execution of this command.

2.2.6.6.8. Quit Command

The 'quit' command is equivalent to the 'exit' command.

2.2.6.7. Help Command

The 'help' command presently only directs the user to type a '?' for help, as shown below.

```
VG> help
        try "?" for help
```

3. Software Configuration

This section discusses the software configuration of the DVG.

3.1. DVG Firmware ROMs

The DVG firmware is contained in two 1 Mbit EPROMs. The two chips bear labels similar to those shown below:

```
  DVG        DVG
VER X.YZ   VER X.YZ
  HI         LO
```

where X.YZ is the current version number and HI and LO refer to the upper and lower bytes of the two byte width. The 'HI' chip is inserted into empty socket U16 and the 'LO' chip is inserted into empty socket U18 on the CPU board

3.2. MVME147 Firmware Configuration

The MVME147 CPU debugger, 147Bug, must be configured for use in the DVG. To obtain a full description of the commands available from the debugger, refer to the MVME147Bug User's Manual. The following procedure can be used to configure an MVME147 as it comes from the manufacturer.

When the debugger is first brought up following a power-up or reset, the following is displayed on the console port:

```
Copyright Motorola Inc. 1988, All Rights Reserved
VME147 Monitor/Debugger Release 2.0 - 3/16/89
FPC passed test
MMU passed test
COLD Start
```

followed by a rapid display of test result messages at the bottom of the screen. Press the black 'ABORT' button on the front panel of the MVME147 to obtain the following menu:

```
1) Continue System Start-up
2) Select Alternate Boot Device
3) Go to System Debugger
4) Initiate Service Call
5) Display System Test Errors
6) Dump Memory to Tape
Enter Menu #:
```

Type a '3' followed by a 'Return' to obtain the diagnostic prompt ('147-Diag>'). Then enter the 'env' command to set the operating environment to 'Bug' as shown below. On the first question respond with a 'b'. On all the following questions, respond with a 'Return' to select the default.

```
147-Diag>env
Bug or System environment [B,S] = S? b
SYSTEM V/68 or VERSAdos operating system [S,V] = S?
Set VME Chip:
Board ID [0-FF] = $00?
GCSR base address [0-0F] = $0F?
Utility Interrupt Mask [0-FE] = $00?
Utility Interrupt Vector [$20-$3E0] = $0180?
Copyright Motorola Inc. 1988, All Rights Reserved
VME147 Monitor/Debugger Release 2.0 - 3/16/89
```

```
FPC passed test
MMU passed test
COLD Start
147-Bug>
```

The 147Bug prompt should now be displayed on the console as shown in the above example.

Next, configure the firmware to boot from the EPROMs by typing the 'rb' command, as shown below.

```
147-Bug>rb
Boot at power-up only      [Y,N] ? Y n
Enable search of VMEbus [Y,N] ? N n
Boot direct address = $FF800000 FFA00000
147-Bug>
```

Respond with the answers 'n', to boot at both power-up and any reset, 'n' to disable search of the VMEbus, and 'FFA00000' to set the boot address to that of the DVG EPROMs.

At this point, power cycling the system or pressing the reset button should boot the DVG firmware (if the DVG EPROMs have been installed). Refer to the section on 'Boot-up an Initialization' for a description of the boot process.

3.3. DVG Firmware Configuration

The DVG firmware operating parameters may be configured from the console port via the DVG Terminal Interface described in the section titled 'Terminal Interface'. The parameters that may presently be configured are:

Ethernet address for voice traffic

exercise ID

radio id's for each SIMVAD channel

ethernet type

frequency mapping for each SIMVAD channel

voice channel loopback

push-to-talk input checking

speech-valid checking

speech-valid threshold

speech coding rate

4. Hardware Configuration

4.1. MVME147 CPU

The jumper configuration for the MVME147 CPU is shown in Figure 1. This is the standard configuration supplied by the manufacturer.

The DVG EPROMs must be inserted into MVME147 sockets U16 and U18. The DVG HI EPROM goes into position U16 and the DVG LO goes into position U18.

4.2. MVME712M Transition Module

MVME712M Configuration:

The MVME712M adapter is configured as shown in Figure 2. Ports 2 through 4 are configured for connection to a modem while port 1 is configured for connection to a terminal via a straight-through cable with a male connector on the MVME712M end.

Cables are routed from MVME712M connectors J2 and J3 to the P2 adapter module plugged into the P2 connector of the back plane.

4.3. SIMVAD Speech Coding Board

The jumper configuration of the SIMVAD speech coding board is shown in Figure 3.

The SIMVAD board has two jumpers, E1 and E2 that are for manufacturing and development use only. These should be installed on all boards.

The SIMVAD has a single dip switch pack, SW2, which sets the base address of the board in the VMEbus short I/O space. The standard configuration DVG contains two SIMVAD boards for a total of four voice channels numbered 0 through 3. The table below shows how to set SW2 for each of the SIMVAD cards up to a total of eight (although only two SIMVAD cards are used in the DVG).

Card	Channels	Address	SW2 (8 through 1)
1	0,1	0xE000	OFF OFF OFF ON ON ON ON ON
2	2,3	0xE100	OFF OFF OFF ON ON ON ON OFF
3*	4,5	0xE200	OFF OFF OFF ON ON ON OFF ON
4*	6,7	0xE300	OFF OFF OFF ON ON ON OFF OFF
5*	8,9	0xE400	OFF OFF OFF ON ON OFF ON ON
6*	10,11	0xE500	OFF OFF OFF ON ON OFF ON OFF
7*	12,13	0xE600	OFF OFF OFF ON ON OFF OFF ON
8*	14,15	0xE700	OFF OFF OFF ON ON OFF OFF OFF

*Not currently used.

4.4. VMEBUS Back plane

The table below shows which slots the circuit cards should be plugged into.

SLOT*	BOARD	COMMENTS
1	MVME147 CPU	system controller
2	SIMVAD	channels 0&1, address = 0xE000
3	SIMVAD	channels 2&3, address = 0xE100
4	Empty	
5	Empty	
6	Empty	
7	Empty	
8	Empty	
9	Empty	
10	Empty	
11	Empty	
12	Empty	

*Slot 1 is at the left when facing the front of the Chassis.

Even numbered channels are the top DB9 connector on a SIMVAD board while odd numbered channels are the bottom DB9 connector.

Each slot has 5 jumpers associated with it (to the right of the slot, refer to Figure 4.). The group of 4 jumpers labeled BG pass the bus grant signals through slots that do not have a card installed. The single jumper labeled IACK propagates the interrupt acknowledge signal through slots that do not have a card installed. For the DVG, no slot in the back plane should have jumpers installed in it.

4.5. Rear I/O Connectors

The MVME147M transition module should be installed in place of the two right-most blanks as shown in Figure 5. The remaining 12 blanks should remain installed.

4.6. P2 Adapter

The P2 adapter board is plugged into the rear of connector P2 of slot 1 of the back plane. Ribbon cables supplied with the P2 adapter and MVME712M transition module are routed internal to the chassis as indicated below. Figure 6 shows how the P2 adapter is inserted into the back of the VMEBus (viewed from the rear of the chassis).

MVME712M		P2 Adapter	Size
J2	to	J2	64 conductor
J3	to	J3	50 conductor

4.7. Parts List

Quantity	Part	Manufacturer	Description
1	MVME147-1	Motorola	CPU, VMEBus 25MHz 68030 with 4MB RAM
1	MVME712M	Motorola	transition module & P2 adapter with cables
2	SIMVAD	BBN	speech coding board, two channels
1	Chassis	MUPAC 5098GCE12VK-104	12 slot VME chassis with power cord
2	EPROM	Mitsubishi M5M27C101K-15	1 Mbit, 150 nanosec
1	Cable???		Ethernet transceiver cable

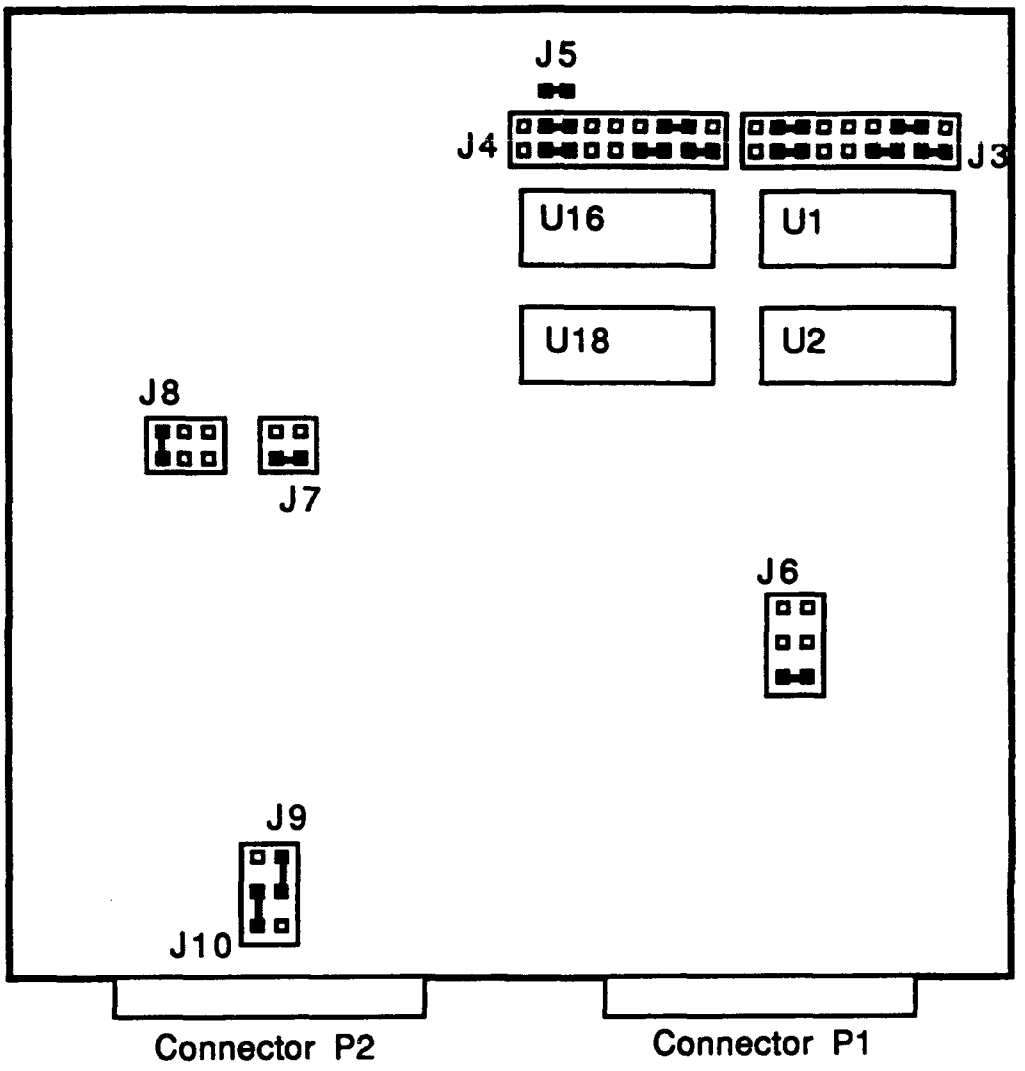


Figure 1. MVME147 configuration. Component side up.

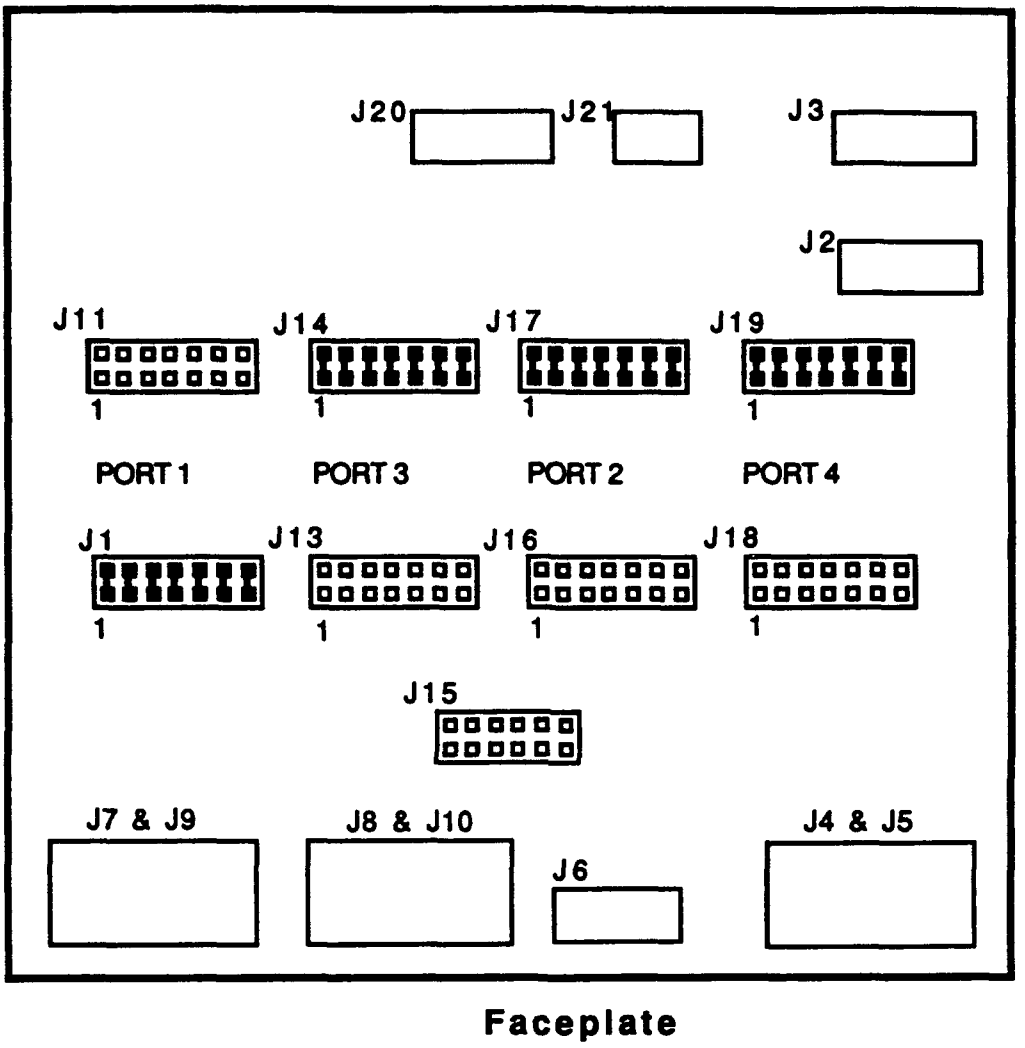


Figure 2. MVME712M Transition Module. Component side up.

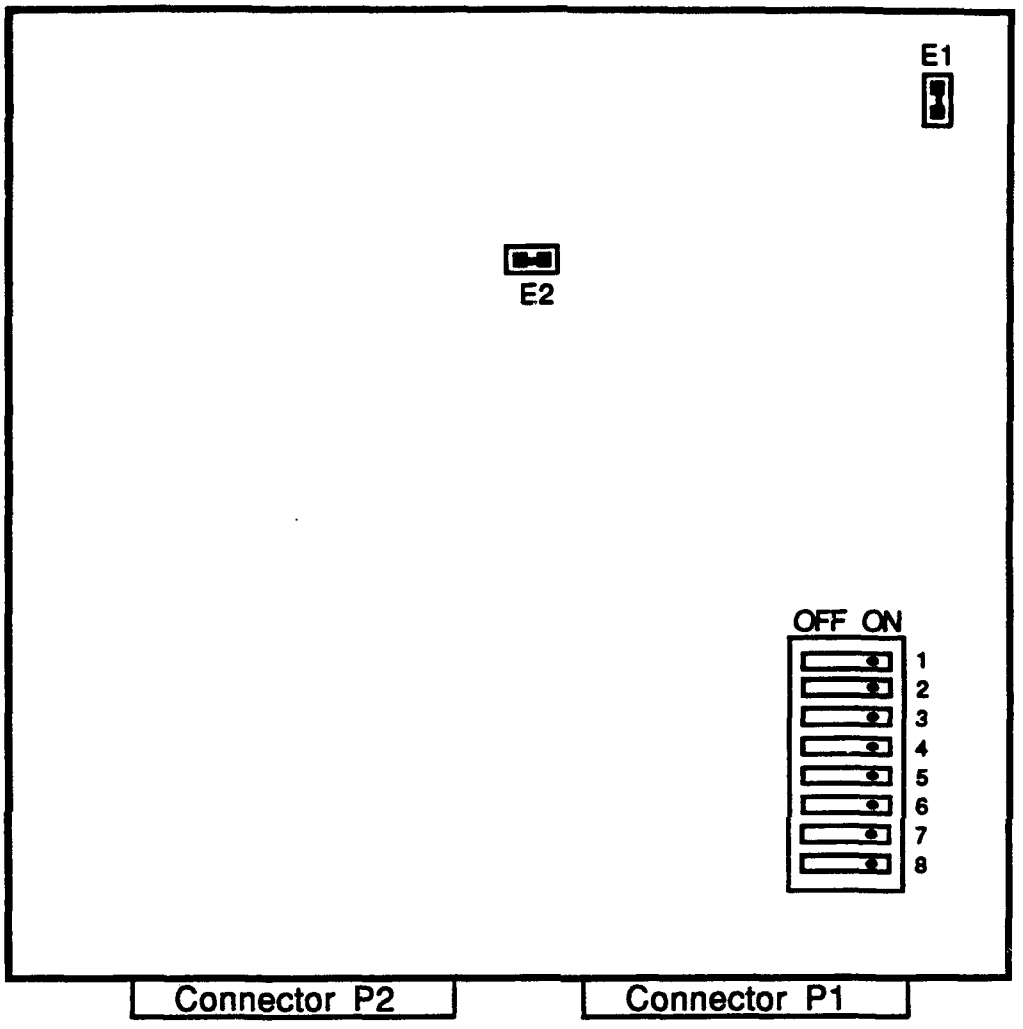


Figure 3. SIMVAD configuration. Component side up.

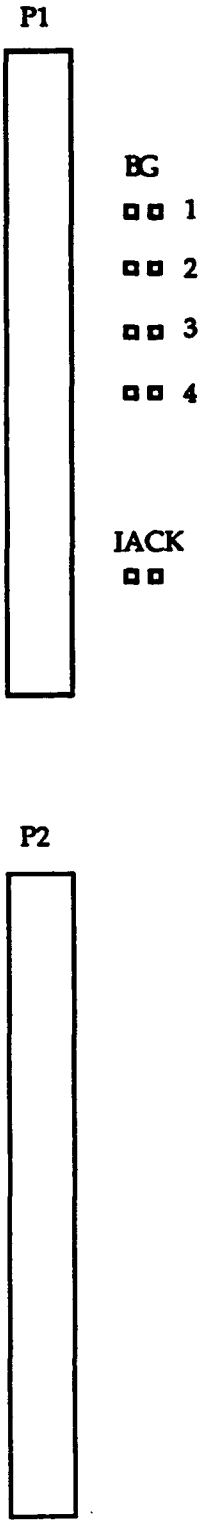


Figure 4. VMEbus Jumpers.

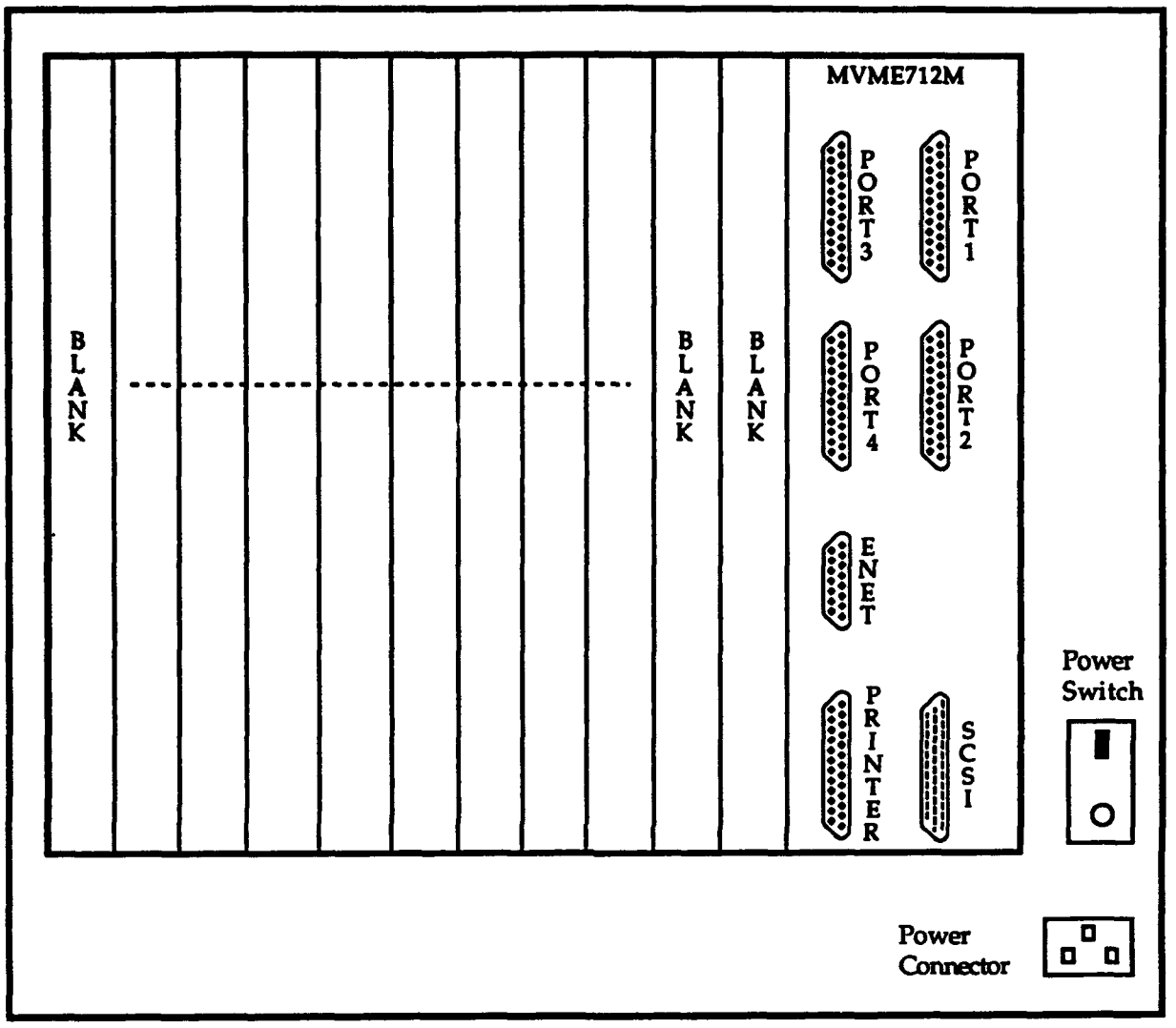


Figure 5. Rear view showing I/O connections.

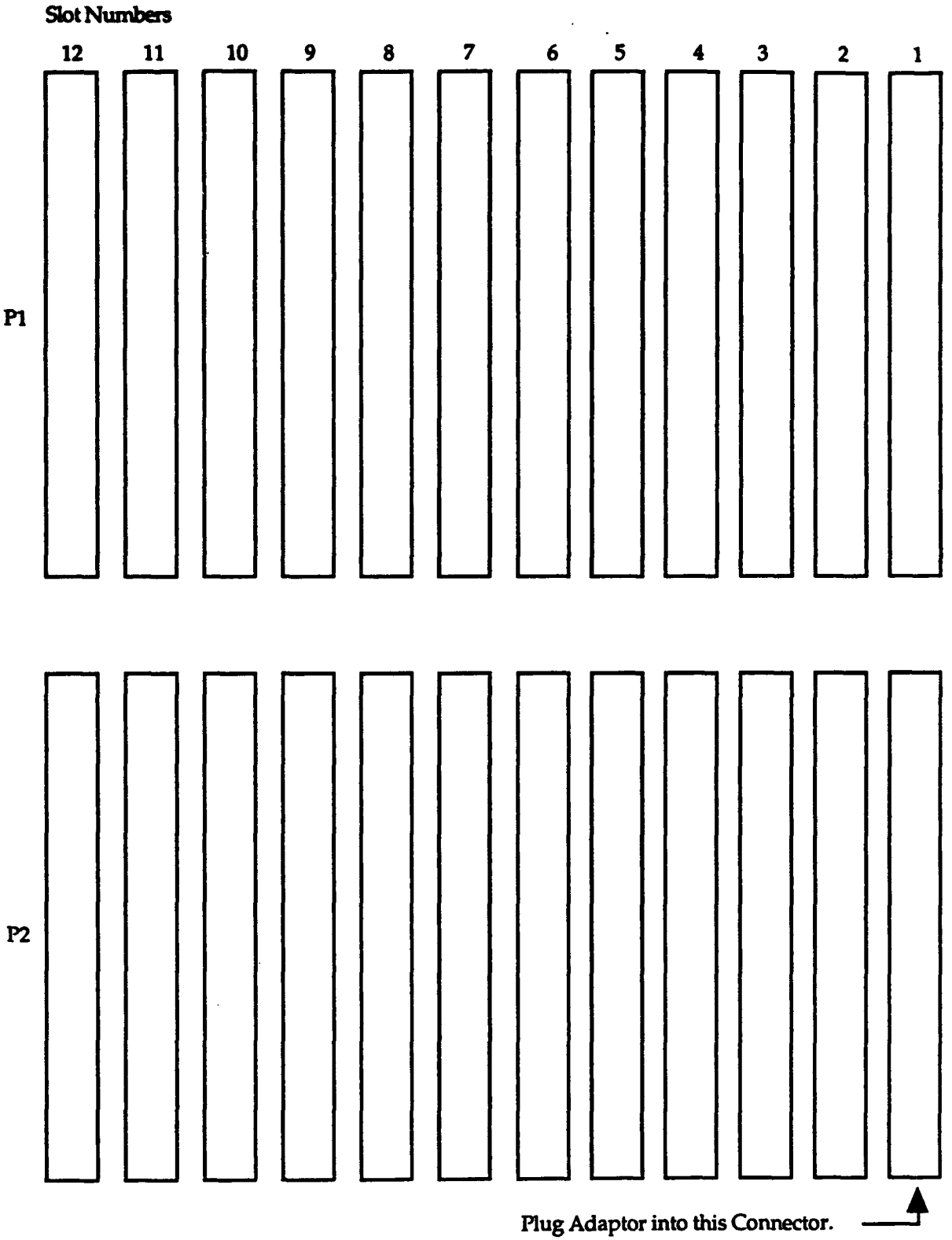


Figure 6. P2 Adaptor Configuration.